**1.INTRODUCTION**

1.1 **Overview**

1. Purpose of the Project
2. Existing Problem
3. Proposed Solution
4. Block Diagram
5. Hardware/Software Designing
6. Experimental Investigations
7. Flowchart
8. Result
9. Advantages & Disadvantages
10. Applications
11. Conclusions
12. Future Scope
13. Bibliography

1.2 **Purpose**

1. Animal detection is done by using smart Analytic device. It verifies the presence of an animal.
2. The Smart device verifies the animal entering into the rural areas.
3. The details of the object are being taken and uploaded into the cloud.
4. We can monitor the entry of an animal and we can make the people in rural areas to be alert by using this IoT device.

**2.LITERATURE SURVEY**

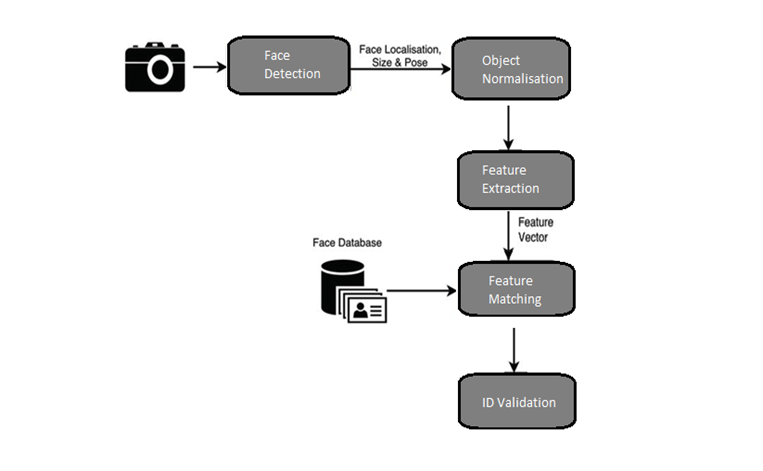
2.1  **Existing Problem**

The problem with the present existing device is it cannot able to identify whether the object detected is animal or human, it just identifies the entry of an object

2.2 **Proposed Solution**

We can make use of IoT Analytics in Animal Detection, such that in rural areas one can identify the object is animal and send an alert message to the forest department and also a play sound to the people in rural areas simultaneously. Also, with the usage of IoT, automatically, thedetails of the person are taken an we can restrict them.

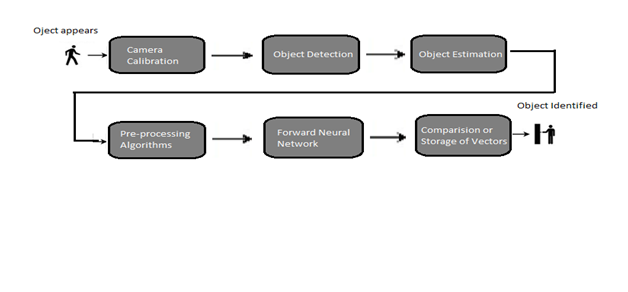
**3.Theoretical Analysis**

**3.1BlockDiagram**

**3.2 Hardware/Software Designing**

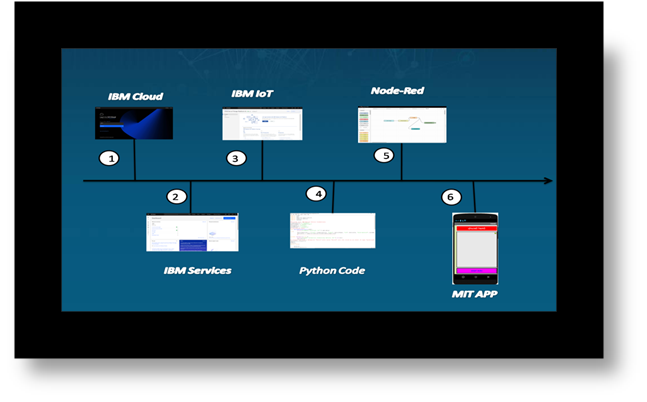
The Software designing involves genera We used IBM Cloud Services to create Cloud storage. In cloud storage we create a bucket. We use these cloud storage credentials in Python program then we make use of the Node-Red platform to display the image.With the help of MIT APP Inverter we designed the app &

integrated with the Node-Red to observe the object.

**4.Experiment Investigation**

To complete our project work we collected the required data from Google & research papers. After getting the complete knowledge we work according to our roles in the project. At first we create the IBM Cloud account then we created the Cloud storage service after we wrote a python code in IDLE to connect Cloud storage. Next we created the Node-Red Services. This service helps us to show virtual flow graphs. From Node-Red we send image to the MIT APP. From app we can view the details of the object.

**5.FLOWCHART**

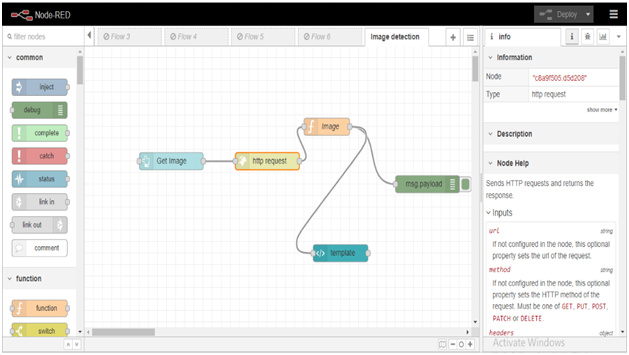


**6 RESULT**

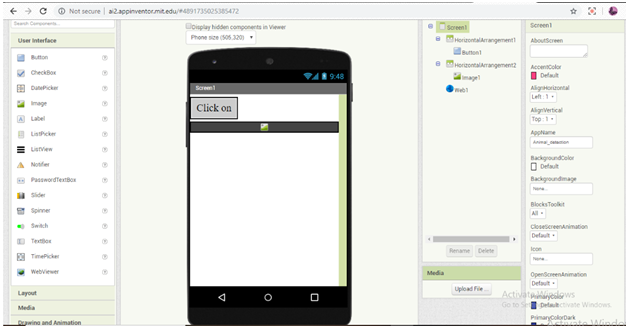
**PythonCode:**



**Node-Red:**



**MIT APP:**



**7 .ADVANTAGES & DISADVANTAGES**

**Advantages:**

1) Increase ease of control in rural areas

2) Keep track of who comes and goes

3) Protect against animals

4) create safe Environment

5) Reduce Accidents and Deaths

6) Easy Monitoring

**Disadvantages:**

1) Animal detection systems can be hacked.

**8. APPLICATIONS**

1) Rural areas.

2) Government Sectors.

**9. CONCLUSIONS**

Regarding the camera and its calibration, the industrial camera had a better performance compared to the webcam as the calibration method presented focus on the best face image that can be acquired. As for software, the detection algorithm presented a good performance.

**10. FUTURE SCOPE**

The future work goes through the implementation of the solution in rural areas where officers and people would use it. Until then, the training of new neural networks using the preprocessing techniques is presented, and the study of new alternatives for cameras is on the agenda.

**11. BIBLIOGRAPHY**

<https://cloud.ibm.com/registration>

<https://cloud.ibm.com/catalog/services/watson-studio>

[http://Ai2.appinventor.mit.edu](http://ai2.appinventor.mit.edu/)

<https://flows.nodered.org/node/node-red-dashboard>

<https://developer.ibm.com/recipes/tutorials/ui-dashboard-for-iot-device-data-using-node-red/>

<https://appinventor.mit.edu/>

**APPENDIX**

**A. Source Code**

import cv2

import numpy as np

import datetime

import ibm\_boto3

from ibm\_botocore.client import Config, ClientError

import time

#CloudantDB

from cloudant.client import Cloudant

from cloudant.error import CloudantException

from cloudant.result import Result, ResultByKey

import requests

from ibm\_watson import TextToSpeechV1

from ibm\_cloud\_sdk\_core.authenticators import IAMAuthenticator

from playsound import playsound

import json

from watson\_developer\_cloud import VisualRecognitionV3

# Constants for IBM COS values

COS\_ENDPOINT = "https://s3.jp-tok.cloud-object-storage.appdomain.cloud"

#Current list avaiable at https://control.cloud-object-storage.cloud.ibm.com/v2/endpoints

COS\_API\_KEY\_ID = "veZRw9HMG\_joB\_PNWfbkwo57R9jI0p5kvY8wrst\_HAFl" # eg "W00YiRnLW4a3fTjMB-odB-2ySfTrFBIQQWanc--P3byk"

COS\_AUTH\_ENDPOINT = "https://iam.cloud.ibm.com/identity/token"

COS\_RESOURCE\_CRN = "crn:v1:bluemix:public:cloud-object-storage:global:a/40c46a853021447092943d3bf060fe66:0cf56133-1937-4855-af03-2f9be53189ab::" # eg "crn:v1:bluemix:public:cloud-object-storage:global:a/3bf0d9003abfb5d29761c3e97696b71c:d6f04d83-6c4f-4a62-a165-696756d63903::"

# Create resource

cos = ibm\_boto3.resource("s3",

ibm\_api\_key\_id=COS\_API\_KEY\_ID,

ibm\_service\_instance\_id=COS\_RESOURCE\_CRN,

ibm\_auth\_endpoint=COS\_AUTH\_ENDPOINT,

config=Config(signature\_version="oauth"),

endpoint\_url=COS\_ENDPOINT

)

def create\_bucket(bucket\_name):

print("Creating new bucket: {0}".format(bucket\_name))

try:

cos.Bucket(bucket\_name).create(

CreateBucketConfiguration={

"LocationConstraint":"jp-tok-standard"

}

)

print("Bucket: {0} created!".format(bucket\_name))

except ClientError as be:

print("CLIENT ERROR: {0}\n".format(be))

except Exception as e:

print("Unable to create bucket: {0}".format(e))

create\_bucket("kareti")

visual\_recognition = VisualRecognitionV3(

'2018-03-19',

iam\_apikey='uOIt\_ok57VcZn6xCcZMyEPNHL-VeX96nRnlj4azpxp\_4')

#Provide CloudantDB credentials such as username,password and url

client = Cloudant("45793cc6-68c2-457d-9936-9acd2774d2f0-bluemix", "85d9ae991b298f8f60a5857b8b4201573d2b9ca988c5a2abbf073643ac0562b7", url="https://45793cc6-68c2-457d-9936-9acd2774d2f0-bluemix:85d9ae991b298f8f60a5857b8b4201573d2b9ca988c5a2abbf073643ac0562b7@45793cc6-68c2-457d-9936-9acd2774d2f0-bluemix.cloudantnosqldb.appdomain.cloud")

client.connect()

#Provide your database name

database\_name = "bhavana"

my\_database = client.create\_database(database\_name)

if my\_database.exists():

print(f"'{database\_name}' successfully created.")

#It will read the first frame/image of the video

video=cv2.VideoCapture(0)

while True:

#capture the first frame

check,frame=video.read()

gray=cv2.cvtColor(frame, cv2.COLOR\_BGR2GRAY)

time.sleep(5)

picname=datetime.datetime.now().strftime("%y-%m-%d-%H-%M")

cv2.imwrite(picname+".jpg",frame)

#waitKey(1)- for every 1 millisecond new frame will be captured

Key=cv2.waitKey(1)

if Key==ord('q'):

#release the camera

video.release()

#destroy all windows

cv2.destroyAllWindows()

break

with open('./'+picname+'.jpg', 'rb') as images\_file:

classes = visual\_recognition.classify(

images\_file,

threshold='0.6',

classifier\_ids='Team09\_657499471').get\_result()

a=json.dumps(classes, indent=2)

b=json.loads(a)

print(b)

c=b['images']

for i in c:

for j in i['classifiers']:

k=j['classes']

for l in k:

print(l['class'])

x=l['class']

def multi\_part\_upload(bucket\_name, item\_name, file\_path):

try:

print("Starting file transfer for {0} to bucket: {1}\n".format(item\_name, bucket\_name))

# set 5 MB chunks

part\_size = 1024 \* 1024 \* 5

# set threadhold to 15 MB

file\_threshold = 1024 \* 1024 \* 15

# set the transfer threshold and chunk size

transfer\_config = ibm\_boto3.s3.transfer.TransferConfig(

multipart\_threshold=file\_threshold,

multipart\_chunksize=part\_size

)

# the upload\_fileobj method will automatically execute a multi-part upload

# in 5 MB chunks for all files over 15 MB

with open(file\_path, "rb") as file\_data:

cos.Object(bucket\_name, item\_name).upload\_fileobj(

Fileobj=file\_data,

Config=transfer\_config

)

print("Transfer for {0} Complete!\n".format(item\_name))

except ClientError as be:

print("CLIENT ERROR: {0}\n".format(be))

except Exception as e:

print("Unable to complete multi-part upload: {0}".format(e))

multi\_part\_upload("kareti", picname+".jpg", picname+".jpg")

json\_document={"link":COS\_ENDPOINT+"/"+"kareti"+"/"+picname+".jpg"}

new\_document = my\_database.create\_document(json\_document)

authenticator = IAMAuthenticator('qiXzgBiIKN32QXgXegbxJ3-OzPMh4eDeEeuEQyaO0eUj')

text\_to\_speech = TextToSpeechV1(

authenticator=authenticator

)

text\_to\_speech.set\_service\_url('https://api.au-syd.text-to-speech.watson.cloud.ibm.com/instances/4b76ef9d-bc67-4762-9fca-7169c9140d05')

with open('kareti1.mp3', 'wb') as audio\_file:

if(x=="Humans"):

audio\_file.write(

text\_to\_speech.synthesize(

f' the object is {x}',

voice='en-US\_AllisonVoice',

accept='audio/mp3'

).get\_result().content)

else:

audio\_file.write(

text\_to\_speech.synthesize(

f' the object is {x}',

voice='en-US\_AllisonVoice',

accept='audio/mp3'

).get\_result().content)

import requests

r=requests.get('https://www.fast2sms.com/dev/bulk?authorization=F5VGTe8iUzn6S2PfRLXD0B9gY4haowsEcyK7MZmxqpk3luCdAOVbwHK27aNkrS5cIqOztxlT9MsdvCfR&sender\_id=FSTSMS&message=Animal is detected in cameras&language=english&route=p&numbers=9182382058')

print(r.status\_code)

playsound('kareti1.mp3')